In Tin Lee, K. & Mitchell, K. (Eds.) *Proceedings to ICCE 2003, Hong Kong, ISBN: 962-949-144-3.* Norfolk, VA: AACE. pp. 556-563.

XMendeL – A web-based semantic Web Tool for e-Learning Production Processes

Ronald Hartwig, Michael Herczeg, Lia Hadley
Institute for Multimedia and Interactive Systems – University of Luebeck
Hartwig|Hadley|Herczeg@imis.uni-luebeck.de

Abstract: The production process of multimedia learning material includes many participants, tasks, and tools. The system described in this paper combines all data from the development and production processes into one semantic web based and XML-coded system. The contents are kept using object-oriented development concepts, such as "inheritance", to cope with large volume and complex data.

Introduction

This paper results from work experiences over four years of learning module production. The authors participate in several projects dealing with virtual universities: in particular, the German flagship project "Virtual University of Applied Science" (in German "VFH – Virtuelle Fachhochschule"), the project "medin" (Multimedia-Based Distance Education in Medical Computer Science) and the project "WissPro" ("Wissensprojekt 'Informatiksysteme im Kontext'"). All projects are sponsored by the German Federal Ministry of Education and Research (BMBF). The goal of these projects is to build interactive and multimedia learning modules for (distance) learning at universities using the Internet. A system was created to support the development and production processes, which follows the principles of a comprehensible quality assurance system. This enables all of the participants to make their decisions and contributions based on a shared process repository.

Concepts

Today's authoring systems (e.g. HTML-editors, XML-tools, or other, so called "authorware") focus mainly on one part of the production process. They are intended for implementing a learning or a teaching module after the concept and course material is complete. The majority of process-relevant information is documented and communicated using other tools: text editors or word processors. Typical process information includes: user descriptions, didactical concepts, use scenarios, and quality requirements.

Theoretically, XML has the potential to combine all process-relevant data into a unified database. Ideally, all information (i.e. context information and content data) is meshed in a semantic web. The meshing process though should remain a creative process for human experts in combination with a content management system. The context information (e.g. didactical, technical, content, usability or organizational related information) must be available during this creative process in order to make conceptual decisions.

The classical gap which often exists between analysis data and actual system design is well known in the world of software engineering. Integrated development environments are intended to overcome the discrepancies which arise because of the gap in the information flow. The first step is to offer a common database for all participants and process phases.

Process Model

The underlying process model for our system is adapted from the iterative ISO 13407 (1999) process approach to fit into the CSCL production context. It is combined with a quality assurance model (see Dzida and Freitag 2001) for usability which is expanded to focus on didactical and technical fields of interest. The following approach is strongly influenced by the scenario-based design approaches used in the field of software engineering (see Holtzblatt & Beyer 1996, Kritzenberger, Hartwig and Herczeg 2001, and Rosson & Carroll (2002)). The central goal of the process model is to implement a consistent decision chain (Hartwig, Triebe and Herczeg 2002). This chain begins with context information (e.g. use scenarios) and ends with product attributes. This makes it possible for the development team (e.g. content authors, didactical experts, usability specialists, producers, and quality assurance managers) or external evaluation persons to track any product attribute back to its original concept and contextual foundations. It also encourages all participants, each with different expertise, to understand why something is required or vital to the development or production processes.

If a specific didactical approach is chosen (e.g. an "explorative model") this leads to a concrete design strategy on how to structure the information blocks and how to connect them. This may be specified as a requirement (i.e. "users are able to navigate directly from one block to any other block"), which can be later checked in a quality assurance test. Following the ideas of usability, the next step is to derive a minimum level of effectiveness, efficiency and satisfaction. In this example a minimal requirement would be that the software solution is less complicated and time consuming than the paper-based method. The next step is to define testable criteria to assure that the technical implementation does not contradict the original concept. These criteria can then be used to setup a production guideline and to do quality assurance management.

The goal of a holistic system is to utilize all the process data to communicate between different participants in the development and production processes and to document the quality management relevant decisions in a central database.

Combining Content and Context Objects in a Unified Database

The different types of information which evolve during the development process can be seen as information "objects" or knowledge "frames" described in (Hartwig, Kritzenberger and Herczeg 2000) and (Kritzenberger & Herczeg 2001). These objects have attributes similar to those from the object-oriented software development world. For example, a user description has user attributes like age, disposition, goals or skills. A development decision (e.g. didactical) has a reasoning and links to pertinent information; e.g. an attribute of the user description or to resulting requirements for the design. The interconnectivity logic of all these objects creates a semantic web.

The objects build the data base and the connecting links carry implicit or explicit relational information (e.g. "this observation leads to this decision" or "this product attribute contradicts the criteria of …"). If an object is a specialization of a more general object this relation is called a "parent-child" relation and the parent may propagate its information to its children ("inheritance").

The goal of our system is to support building such objects into a semantic web (Kritzenberger & Herczeg 2001). In practice, the structure and the contents of this semantic web are not always well defined at the beginning of the process and it evolves during the entire lifecycle of the learning module. Attributes are added as soon as analysis (e.g. an evaluation review) discloses them or the production process needs them. New types of connections appear when the different participants review and rereason their previouslymade decisions. Therefore a supporting software system has to allow the adaptation of the structure and the contents of the semantic web by its users throughout the development and production processes.

Modeling the context information in one system and modeling the content data in another deepens the gap between analysis and design. The specialized supporting experts mainly use different forms of text formatting systems in order to keep their context information; while producers use content objects like HTML-pages or Macromedia Flash™ elements. Context information and content data have to be unified and connected. There already exist standards, such as the IEEE Learning Objects Metadata (2001), to model content data. The system must be able to take such external standards in consideration. Figure 1 shows the main structure of the semantic web structure. In practice, the first iteration of such a process is based only on rough assumptions and common knowledge instead of detailed analysis data. The underlying

process model tries to overcome this potential drawback. The evaluation phase is taken as a second analysis phase as well. Valuable use data is gathered during on-site usability testing as described in (Dzida & Freitag 2001) and fed back into the next iteration. An integrative tool supports this dynamic process; making it possible to trace the effect of each change in the context of use, down to the product design decisions.

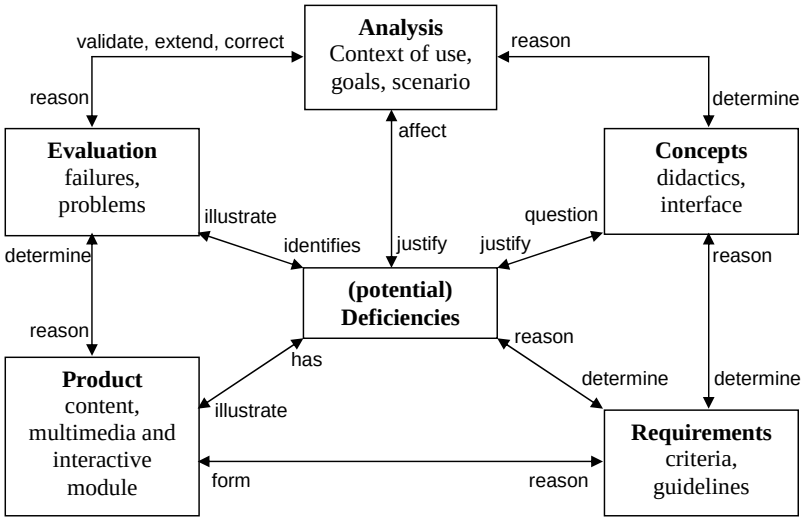

Figure 1: Object model with its interconnectivity

The process previously described is based on a falsification approach. It is assumed in this approach that as long as no specific fault has been determined, the product quality is acceptable. A falsification approach avoids wasting time and money on worst-case scenarios. In our system the majority of process context information is connected to potential deficiencies as a consequence of the falsification approach. Potential deficiencies are identified during evaluation processes and reviewed using their context information. The original concept decisions and design rationale related to these potential deficiencies must first be evaluated before they can be called deficiencies.

Backtracking and Tracing

One of the main goals of our system is to support the backtracking of decision chains. This allows critical review of the decisions and their compliance to system requirements. All links within this semantic web are automatically bidirectional in order to support backtracking. If, for example, a product attribute conflicts with a specific criteria then this product attribute illustrates the contra-meaning of the criteria. Additionally, if, for example, a decision is based on a user description, it is vital to be able to comprehend the effects of a change in this user description.

Using an Inheritance Mechanism

Generally, the number of objects grows rapidly if such an object-oriented model is applied. The implicit inheritance of a hierarchical structure is supported explicitly by the system to cope with the rising complexity of objects and their interconnection. If information is defined on a higher abstraction level ("parent" object), then it must be made available to all dependents ("children"). If an exercise, for example, is available for a whole chapter (parent), and this chapter consists of subchapters and single pages (children), then the link to this exercise is made available even on the single page level. Typical XML-tools support this implicitly as they allow building tree structures. The disadvantage of implicit inheritance is that the user has to manually find the information defined on higher abstractions levels. This leads to a

permanent source of error if the semantic web has a large number of levels which are widely spread. Users may lose track of what attribute is valid on a lower abstraction level.

In a complex and long-termed production process it can not be assumed that the participants have a thorough understanding of information structures, implicit inheritance, and abstraction levels. Our principle use case is a long-term production process (3-5 years of production) with approx. 120 participants. Therefore our system supports an adaptable explicit inheritance mechanism. “Explicitly” means that the inherited data is not only linked to the currently viewed object, but automatically augments the object’s attributes. This makes the object self-contained and the users do not have to understand or follow the hierarchy if they do not want to. There are two forms of inheritance available in our system: one which augments information from parent to child, and the other which replaces information. Replacement is needed for internal attributes like access rights. Most of the information though is an accumulation of all preceding attribute information from higher levels (parents): e.g. a user description may be specialized for subgroups where the general attribute for all users is true, as well as the more specific information for a subgroup. Inheritance can also be made inactive for certain parts of the hierarchy and certain attributes.

Support for all Participants

The initial impulse to develop our integrative system was to overcome problems with the low efficiency of a document-based quality management and decision processes in the German flagship project “Virtual University of Applied Science”. If the producers had questions about how to implement certain product attributes, the support department (didactics and ergonomics) had to write individual reports reasoning their decisions. If, years later, another team needed assistance with a similar problem, it was often difficult to retrieve the older decision. When the first evaluation results from the usability testing were available, changes in the criteria and in many basic assumptions evolved. It was not obvious to determine if and how past decisions were affected by these changes.

In the project “medin” the production process is based on Microsoft Word™ documents from the content authors, which were originally transformed into web-sites using HTML-Editors by the producers. The content authors know little about HTML-technology and therefore they are not able to implement their final changes to the learning modules without the assistance of the producers. Since the content authors and producers are geographically separated, communications of future content changes is often time-consuming. It became evident that our system must enable all participants to easily contribute information with respect to their area of competence. Another system requirement was to make the decisions and their interconnectivity transparent.

Implementation

The system we developed is based on the assumptions and requirements described above. The main principle is to use the XML-meta language as the data modeling base for context information as well as content data: thus bridging the previously mentioned gap in information flow. Unlike pure XML-editors, our system combines XML-transformation (similar to a XSLT processor) with explicit inheritance. This system core is called “XMendeL”: a combination of “XML” and the name of the discoverer of inheritance “Mendel”. Figure 2 shows the main system components of XMendeL. It contains a standard SQL-database (mySQL) which stores the XML-tagged attribute data (context information as well as content data) and their interconnectivity. Binary data (e.g. pictures, video clips), word-processor-documents, interactive contents is stored in a central resource repository on the server file system. Each binary file is accessed and administrated through a respective object in the database. The database is connected to a JAVA-Servlet-Server (Tomcat) via JDBC. The business logic is implemented as a relatively small set of java-servlets.

This application core of XMendeL offers the central functionality of generating self-contained objects in context specific views. The application first reads a set of cascading templates (similar to CSS) which define the appearance and the inheritance strategy of the specific view. The template’s transformation and inheritance rules are attribute dependent and may vary from view to view. Then links between objects and their corresponding back-tracking information are added. In the third step, built-in or user-defined plug-ins are executed. One example of a context specific plug-in is one which automatically

generates page numbering following a dedicated learning path. The final step is to transform XML into the target language (another XML-dialect, HTML, LaTeX, plain text, etc.). The actual procedure is more complex. The resulting output is then made available to all the participants using three different interfaces: a standard web browser, import/export, and web-services.

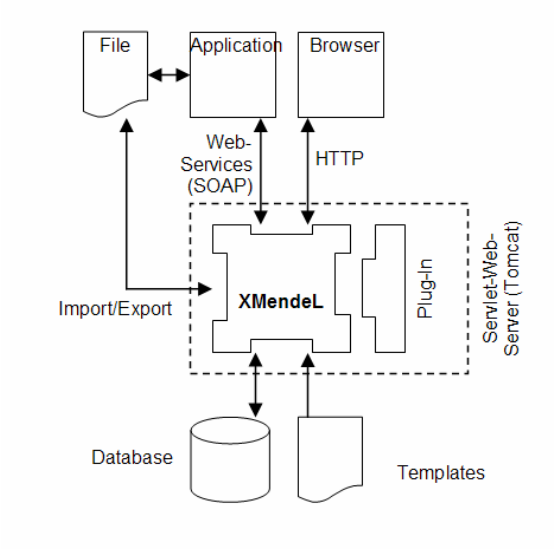

Figure 2: Main tool structure

Web-Browser Access

The standard HTTP-service allows internet browsers to access and administrate the database using a comfortable web front end (see Fig. 3 and 4). The input interface uses special browser extensions to offer a WYSIWYG-feeling for textual formats. Java-scripting is used to implement other dynamic effects like a drag-and-drop interface for object hierarchies or graphical elements. Content authors are offered this simple WYSIWYG-view to read and edit their material. The conception and production staff receives an augmented version with context information (e.g. inherited meta-data and/or storyboards). Quality management staff is provided with yet another view; which clarifies how the decisions were made, and how these decisions are dependent on the analysis data.

File Import/Export

The system can generate a standalone version of an object hierarchy including all needed resources. This is a vital option for learning contents because this way the produced modules may be used in combination with common learning online environments (e.g. Blackboard™). Normally, the stand-alone version is a self-contained HTML-site. Another possibility is to save the data as a XML-conform structure that may be reused by other tools. A third alternative is to export online-contents as a LaTeX-file and then use the LaTeX-typesetting mechanisms to achieve a better printing layout. Theoretically, all SGML-successors may be exported. A practical advantage of this offline file generation option is the user's ability to re-use system data with other external tools. If external tools create well-defined and structured output files, they may be imported into the database using a similar mechanism as described above. Specific import filters are used in our projects to import lecturing presentations generated with HTML-Editors, as well as statistical evaluation data from Microsoft Excel™-sheets. The export/import function allows users, on the one hand, to export portions of the database into a file. They edit the file using external tools and then re-import it again into the database after-wards.

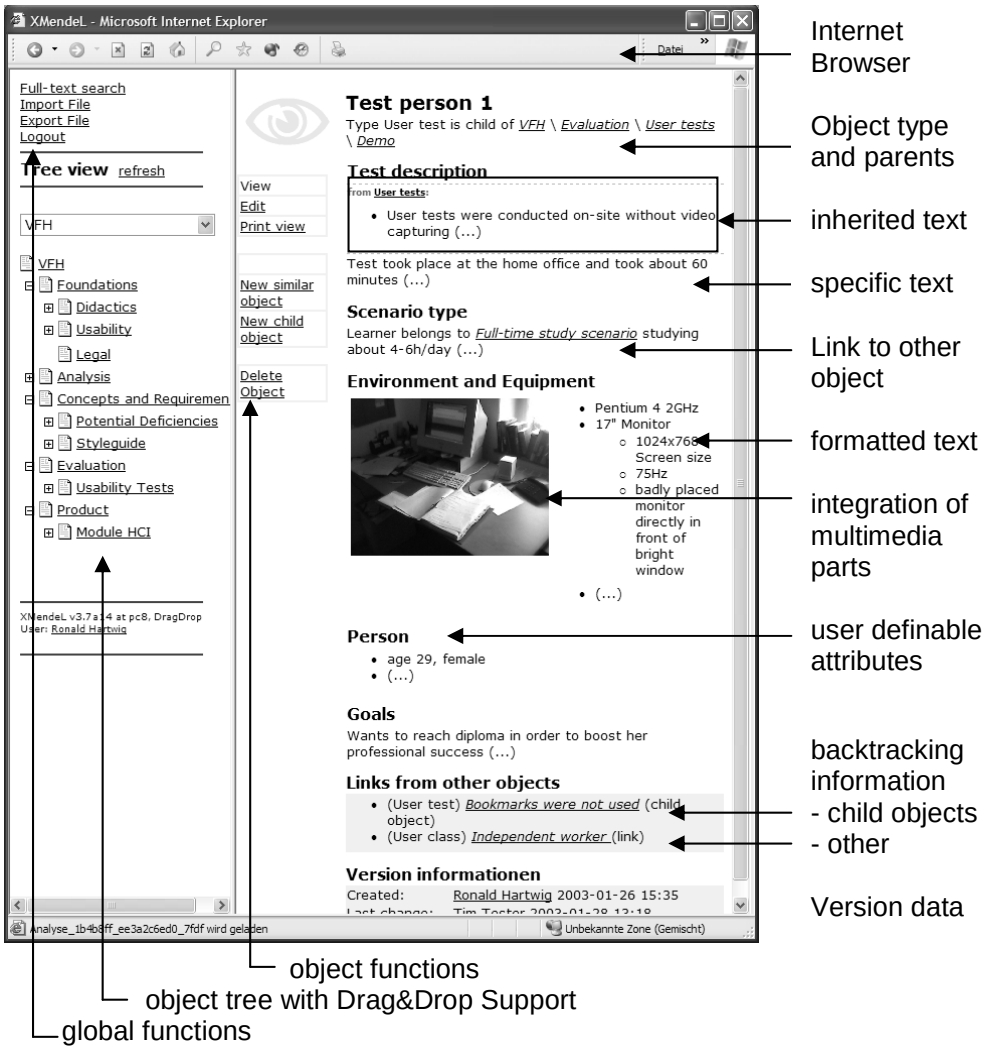

Figure 3: Web interface with standard objects

Web-Services

The third possible interface is the new web services interface using the SOAP-standard. This enables remote access to the XMendeL functionality and database for external applications. A seamless combination of Microsoft Word™ and XMendeL is currently under development. This will allow the authors the ability to continue using their preferred text processor. They do not have to learn how to handle objects, structures and meta-data (e.g. XML-tagging). This integration using SOAP and the Microsoft-.net™-initiative will help participants to easily start using the system; raising the acceptance and keeping the database consistent with the context information and content data. Other participants will continue to access the database using the web-based interface described above.

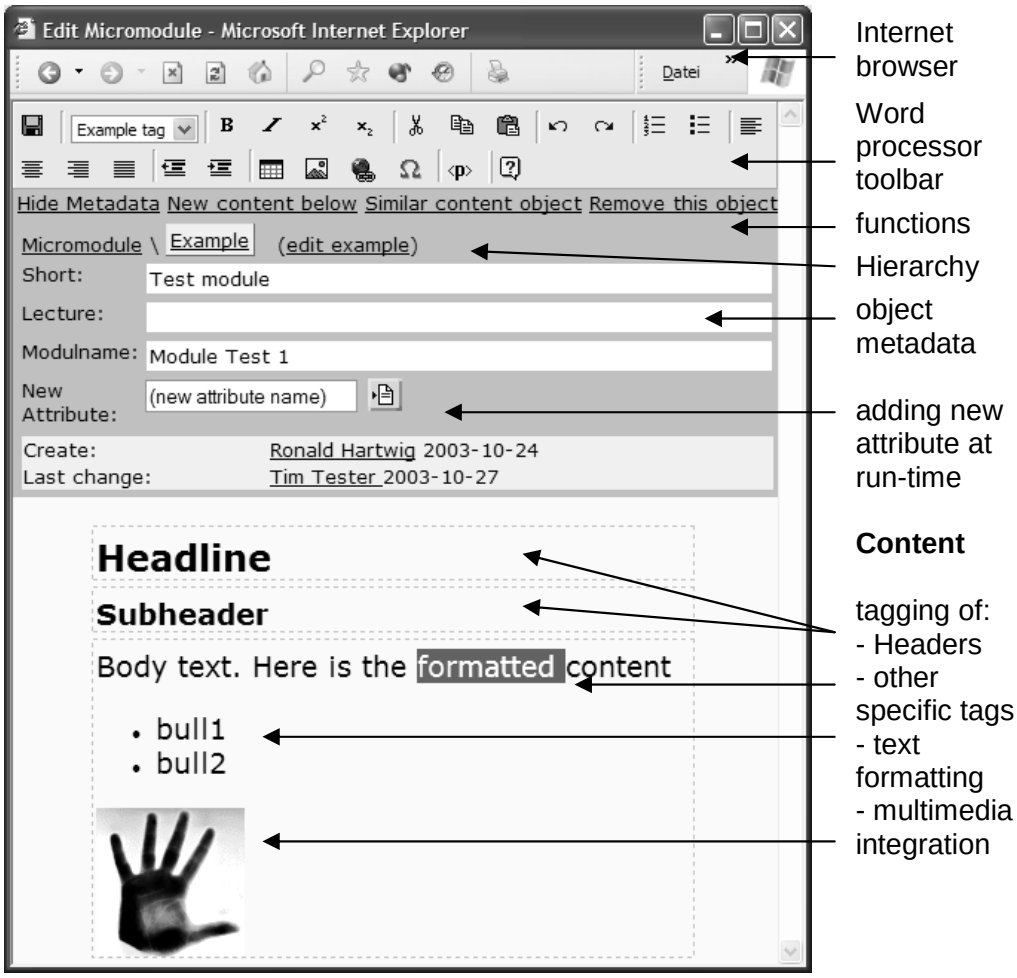

Figure 4: Content specific WYSIWYG-interface

Outlook

The system is currently intended as an information system for human experts in combination with a content management system. The transformation of content data to the resulting product is an automatic process (e.g. XML to HTML). It is the question whether or not development, design, quality management, and other decision-making processes might be automated as well. The open plugin-interface theoretically could be used to implement such a holistic automation process. The major concern would be that such a process would need a more formal and less flexible semantic web and the transformation rules would require extensive use of artificial intelligence.

Summary

Currently the system is used primarily to support the work of usability engineering for learning modules. The system's database contains approximately 1400 context information objects (400 didactical, technical and ergonomic requirements objects, and 1000 evaluation objects), and 5000 content data objects

from learning material. Using bidirectional links and inheritance proved helpful to structure and classify the evaluation results and to connect them to the project guideline. The decision review process runs more efficiently compared to the text-processor approach. Additional engineering is being done to optimize and adapt the views and the user interfaces to the needs of the process participants and to offer it to a broader project audience. The authors believe that a unified semantic web approach as described in this paper is an efficient alternative when working on a complex task such as learning material development with a large variety of human expertise.

References

Dzida, W., Freitag, R.: Usability Testing - The DATech Standard. In: Wieczorek, Meyerhoff (Editor): Software Quality - State of the Art in Management, Testing And Tools. Springer, New York, 2001

Hartwig, R.; Kritzenberger, H.; Herczeg, M.: Course Production Applying Object Oriented Software Engineering Techniques. In: Proceedings of ED-MEDIA 2000, AACE, Canada, 2000

Hartwig, R.; Triebe, J.K.; Herczeg, M.: Usability Engineering as an Important Part of Quality Management for a Virtual University. In: Proceedings of Networked Learning 2002 ICSC-NAISO Academic Press, Canada/The Netherlands. 2002

Holtzblatt, K.; Beyer, H.: Contextual Design: Principles and Practice – In: Field Methods for Software and Systems Design. D. Wixon and J. Ramey (Eds.), John Wiley & Sons, Inc., New York, USA, 1996

IEEE: P1484.12/D6.1 - Draft Standard for Learning Object Metadata “LOM”, IEEE New York, USA, online: http://ltsc.ieee.org/ 2001

International Organization for Standardization: ISO 13407 - Human-centred design processes for interactive systems. International Standard, 1999

Kritzenberger, H.; Hartwig, R.; Herczeg, M.: Scenario-Based Design for Flexible Hypermedia Learning Environments. In: Proceedings of ED-MEDIA 2001, Ontario, Canada, 2001

Kritzenberger, H.; Herczeg, M.: Knowledge and Media Engineering for Distance Education. In: Stephanidis, Constantine (Ed.): Universal Access in HCI. Towards an Information Society for All. Volume 3 of the Proceedings of HCI International 2001. Mahwah, New Jersey, London: Lawrence Erlbaum Associates, 2001, pp. 827-831

Kritzenberger, H.; Herczeg, M.: Task-Model Driven Design of Adaptable Educational Hypermedia. In: Proceedings of Web-Net, Orlando, USA, 2001

Rosson, M. B.; Carroll, J.M.: Usability Engineering – Scenario based development of human-computer interaction, Morgan Kaufmann Pub.; San Francisco, 2002